



# Computational Geometry: Theory and Applications

[www.elsevier.com/locate/comgeo](http://www.elsevier.com/locate/comgeo)



## Largest bounding box, smallest diameter, and related problems on imprecise points<sup>☆</sup>

Maarten Löffler<sup>\*</sup>, Marc van Kreveld

Department of Information and Computing Sciences, Utrecht University, The Netherlands

### ARTICLE INFO

#### Article history:

Received 4 September 2007

Received in revised form 6 March 2009

Accepted 31 March 2009

Available online 21 June 2009

#### Keywords:

Bounding box

Smallest enclosing circle

Diameter

Width

Data imprecision

### ABSTRACT

Imprecision of input data is one of the main obstacles that prevent geometric algorithms from being used in practice. We model an imprecise point by a region in which the point must lie. Given a set of imprecise points, we study computing the largest and smallest possible values of various basic geometric measures on point sets, such as the diameter, width, closest pair, smallest enclosing circle, and smallest enclosing bounding box. We give efficient algorithms for most of these problems, and identify the hardness of others.

© 2009 Elsevier B.V. All rights reserved.

## 1. Introduction

Given a set of points in the plane, various measures exist that try to capture certain properties of that point set. Examples of such measures include the *diameter*: the largest distance between any pair of points, the *closest pair*: the smallest distance between any pair of points, the *width*: the smallest distance between two parallel lines with all points between them, the smallest circle containing all points, or the smallest axis-aligned bounding box containing all points. All these measures have been well studied and optimal algorithms to compute them are known.

When dealing with real-world data, however, locations of input points are often not known exactly. If we know for each point that it lies inside some region, but not where in that region, it becomes interesting to compute bounds on the possible values of these basic geometric measures. To make this more precise, we are given a set of regions in the plane  $\mathcal{L} \subset \mathcal{P}(\mathbb{R}^2)$  and a measure  $\mu : \mathcal{F}(\mathbb{R}^2) \rightarrow \mathbb{R}$  that takes a set of points and gives a real number, and we want to place one point in each region of  $\mathcal{L}$  such that the resulting point set maximises or minimises  $\mu$ .

We study five basic measures both for maximisation and minimisation, when the imprecise points are modelled as squares or discs, possibly overlapping and of different sizes. Some of these problems have already been studied in other contexts, and efficient algorithms or hardness results are known. For most of the remaining problems, we present efficient algorithms here, varying in difficulty from relatively straightforward to more involved. Table 1 summarises all previous and new results. For the problem of computing the largest possible width we have not found any satisfying result, although we can prove NP-hardness when the imprecise points are modelled as line segments. For the smallest diameter of a set of discs, we have no exact algorithm but a polynomial time approximation scheme.

<sup>☆</sup> This research was partially supported by the Netherlands Organisation for Scientific Research (NWO) through the project GOGO.

<sup>\*</sup> Corresponding author.

E-mail addresses: [loffler@cs.uu.nl](mailto:loffler@cs.uu.nl) (M. Löffler), [marc@cs.uu.nl](mailto:marc@cs.uu.nl) (M. van Kreveld).

**Table 1**

New and known results.

| problem                   | model         | largest       | smallest   |
|---------------------------|---------------|---------------|--|
| smallest bounding box     | squares       | $O(n)$        | $O(n)$   |
|                           | discs         | $O(n)$        | $O(n^2)$   |
| smallest enclosing circle | squares       | $O(n)$        | $O(n)$ [8]   |
|                           | discs         | $O(n)$        | $O(n)$   |
| diameter                  | squares       | $O(n \log n)$ | $O(n \log n)$  |
|                           | discs         | $O(n \log n)$ | $(1 + \varepsilon)$ -approx. in $O(n^{c\varepsilon^{-\frac{1}{2}}})$ |
| width                     | squares       |               | $O(n \log n)$ [20]   |
|                           | discs         |               | $O(n \log n)$  |
| closest pair              | line segments | NP-hard       | $O(n \log n)$ [20]   |
|                           | squares       | NP-hard [6]   | $O(n \log n)$  |
|                           | discs         | NP-hard [6]   | $O(n \log n)$  |

### 1.1. Related work

Data imprecision in computational geometry is often considered in stochastic or fuzzy models [9,21]. However, in recent years there has been a growing interest in exact models of imprecision. Guibas et al. [7] introduce the notion of *epsilon geometry*, a framework for robust computations on imprecise points. Abellanas et al. [1] study the *tolerance* of a geometric structure: the largest perturbation of the vertices such that the topology of the structure remains the same.

Some of the problems we consider here also appear in different settings. Colley et al. [5] compute the smallest area axis-aligned rectangle that intersects a set of convex polygons in  $O(n \log n)$  time. Fiala et al. [6] consider the problem of finding distant representatives in a collection of subsets of a given space. In particular, they prove that maximising the smallest distance in a set of  $n$  imprecise points, modelled as circles or squares, is NP-hard. Cabello [3] gives approximation algorithms for this case. Jadhav et al. [8] consider the *intersection radius* of a set of objects: the smallest circle that intersects them all. Robert and Toussaint [20] develop an algorithm for computing the smallest strip that intersects a set of convex regions, while surveying several facility location problems. Such stabbing and facility location problems are related to the minimisation variants of the problems we study.

Averbakh and Bereg [2] also consider imprecise points. Where we compute the point set for which the smallest enclosing circle is worst, they compute the smallest circle that encloses the worst case point set. They study this problem in a facility location context, and they also consider weighted points and different metrics. Nagai and Tokura [16] compute the union and intersection of all possible convex hulls to obtain bounds on the solution. As imprecision regions they use discs and convex polygons, and they give an  $O(n \log n)$  time algorithm. They also compute (possibly non-obtainable) lower and upper bounds for the diameter.

In [12], we also study a classical geometric problem, the convex hull, in an imprecise context. We measured the size of the convex hull by area and perimeter, and presented algorithms to maximise and minimise these measures. The running times vary from  $O(n \log n)$  for computing the smallest area of a set of imprecise points modelled as parallel line segments, to  $O(n^{13})$  for computing the largest perimeter of a set of imprecise points modelled as unit squares. In [11], we prove that maximising the area of the convex hull of a set of imprecise points modelled as arbitrary line segments is NP-hard, and we give approximation algorithms for some of the harder problems. Kruger [10] studies higher-dimensional versions of some of our problems.

In the remainder of this text, we will treat five different geometric measures, for which we present some algorithms and hardness results. The first algorithms are relatively straightforward, but some of the later results, in particular in Section 4, are considerably more complicated.

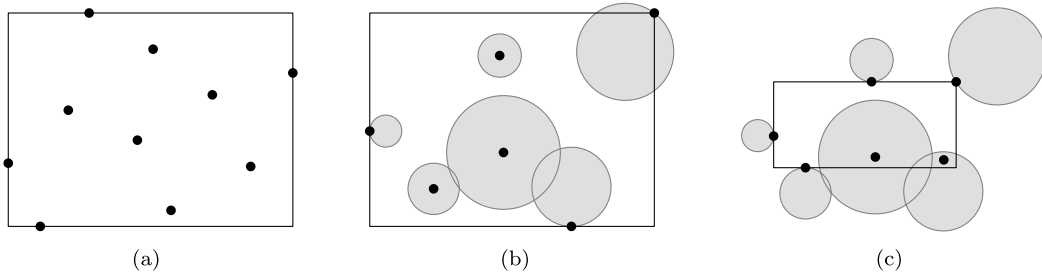
## 2. Axis-aligned bounding box

We start with a relatively simple problem. Given a set of points  $P$ , the axis-aligned bounding box (AABB) is the smallest axis-parallel rectangle that contains  $P$ , see Fig. 1(a). In an imprecise context, we are given a set  $\mathcal{L}$  of regions, and we want to place a point in each region such that the bounding box of the resulting point set is as large or as small as possible, see Figs. 1(b) and 1(c). We will measure the size of a rectangle by its area.

### 2.1. Largest possible AABB

The largest possible AABB can be computed in linear time for both the square and disc model (or, in fact, any other constant-complexity model). Let  $\mathcal{L}' \subset \mathcal{L}$  be the set of the four most extreme regions from  $\mathcal{L}$  in the four axis-parallel directions, making sixteen elements in total. For example, the four topmost regions are the four regions for which their topmost points are the highest four.

**Lemma 1.** *The largest possible AABB of  $\mathcal{L}'$  is equal to the largest possible AABB of  $\mathcal{L}$ .*



**Fig. 1.** (a) The axis-aligned bounding box of a set of points in the plane. (b) The largest possible ABB of a set of imprecise points. (c) The smallest possible ABB of a set of imprecise points.

**Proof.** Suppose this is not the case. Then there is a region  $l$  in  $\mathcal{L} \setminus \mathcal{L}'$  that contributes to the ABB of  $\mathcal{L}$ , say to the top boundary. However, there are at least four regions in  $\mathcal{L}$  that extend higher than  $l$ , of which only three can contribute to another boundary. That means we can place the point of the fourth at its topmost position, and we have a larger ABB. This contradicts the assumption.  $\square$

The ABB of  $\mathcal{L}'$  can be determined by four points each lying on one of the sides of the bounding box, or by only three or two points when one or two points lie on corners. Since  $\mathcal{L}$  has only constant size, we can try all possibilities and report the largest one.

**Theorem 1.** Given a set of  $n$  squares or discs, the problem of choosing a point in each square or disc such that the axis-aligned bounding box of the resulting point set is as large as possible can be solved in  $O(n)$  time.

## 2.2. Smallest possible ABB

The smallest possible ABB is the smallest rectangle that contains at least one point of each region, so it is actually the smallest rectangle that intersects all regions. Let the *left extreme line* be the leftmost of the lines through the rightmost points of all regions. Similarly we define the *right*, *top* and *bottom* extreme lines. When the left extreme line is to the right of the right extreme line, or the top extreme line is below the bottom extreme line, there exists a zero area solution. Otherwise, they define a rectangle  $R$ .

The smallest possible ABB is the smallest rectangle that contains at least one point of each region, so it is actually the smallest rectangle that intersects all regions. Note that this rectangle must contain  $R$ .

### 2.2.1. Squares

When the points are modelled as squares, we know that every square must intersect  $R$ . Therefore,  $R$  is actually the smallest ABB, and we do not need to do anything.

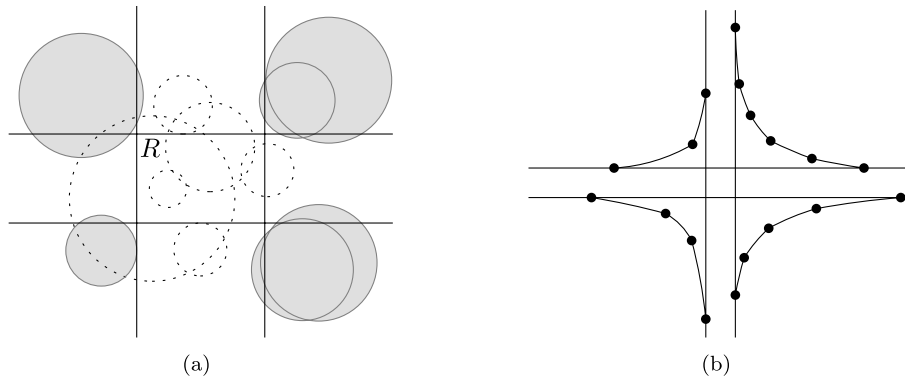
**Theorem 2.** Given a set of  $n$  squares, the problem of choosing a point in each square such that the axis-aligned bounding box of the resulting point set is as small as possible can be solved in  $O(n)$  time.

### 2.2.2. Discs

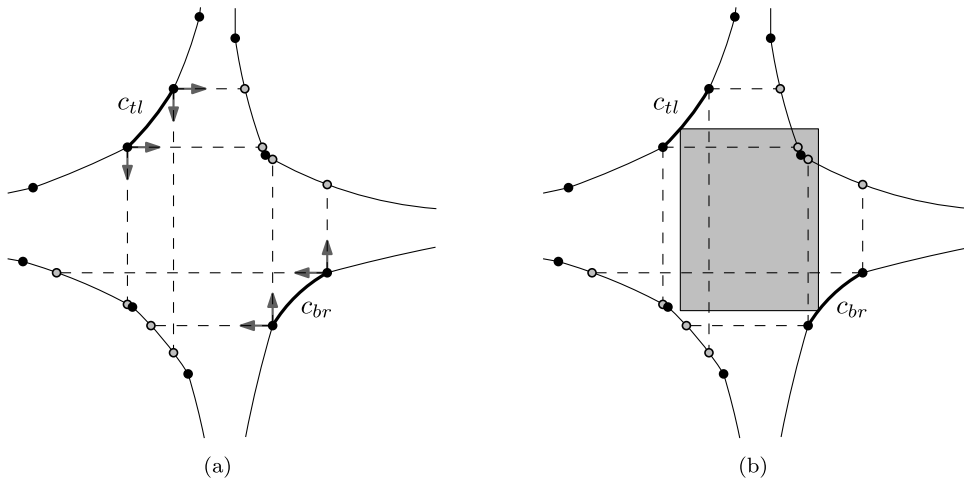
When the imprecise points are modelled as discs,  $R$  does not necessarily intersect all discs, see Fig. 2(a). Colley et al. [5] study the same problem for a set of convex polygons, and they obtain an  $O(n \log n)$  time algorithm. We can use similar techniques to get a quadratic-time algorithm for the disc case.

We do know that  $R$  needs to be contained in the smallest ABB, and therefore we do not need to consider the discs that intersect  $R$  anymore. The centre points of the remaining discs lie outside the two strips between the extreme lines; this partitions them into four groups. Consider the top right group; the other groups are treated symmetrically. For each disc in that group, we define the region where the top right corner of the ABB must lie so that the ABB intersects the disc. Take the bottom left quarter-disc boundary, and extend it to an infinite curve using a half-line vertically upward from the topmost point and a half-line horizontally rightward from the rightmost point. This results in the boundary of a rounded quadrant; the curve divides the locations of the top right corner of the ABB where the ABB intersects this disc, the interior, from those where it does not, the exterior. We compute this curve for each disc in the top right group, and then compute the boundary of the common interior, giving the top right chain. We perform symmetric steps for the other three groups. Each of the four chains is a convex chain of  $O(n)$  circular arcs and two half-lines, see Fig. 2(b). We can compute these chains in  $O(n \log n)$  time by first computing the common intersection of the discs in each group.

The corners of the smallest ABB must lie on or behind those four chains. This means that either the top left and bottom right corners lie on their respective chains, or the bottom left and top right corners lie on their respective chains, otherwise



**Fig. 2.** (a) The discs that intersect the rectangle  $R$  (bounded by the extreme lines) are always accounted for. (b) The remaining discs form four chains of circular arcs.

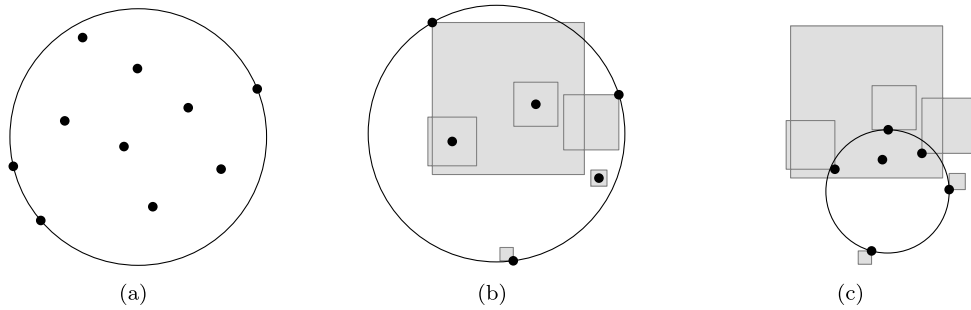


**Fig. 3.** (a) The endpoints of one arc of the top left chain and one arc of the bottom right chain are projected on the top right and bottom left chains. (b) The unrestricted smallest ABB with corners on  $c_{tl}$  and on  $c_{br}$ .

we could shrink the solution. We can try both of these options, so suppose the top left and bottom right corners lie on the chains.

For any pair of a circle segment on the top left chain and a circle segment on the bottom right chain, we can compute the smallest rectangle with a corner on both segments in constant time. However, we need to ensure that the resulting bottom left and top right corners are on or beyond their chains as well. We will maintain the horizontal projection of the endpoints of the circle segment of the top left chain on the top right chain, and the vertical projection of these endpoints on the bottom left chain. In the same way, we keep track of the projections of the endpoints of the circle segment of the bottom right chain on the top right and bottom left chains, see Fig. 3(a).

For every circle segment  $c_{tl}$  on the top left chain, we go over every circle segment  $c_{br}$  on the bottom right chain from left to right, and treat this pair. We first determine the intersections of the projections of  $c_{tl}$  and  $c_{br}$  on the bottom left and top right chains. Observe that the projection of  $c_{br}$  moves only from left to right on the top right chain and only from right to left on the bottom left chain. If the intersection on a chain is empty, say, on the top right chain, then there are two possibilities: either (i) there is no rectangle with its top left corner on  $c_{tl}$  and its bottom right corner on  $c_{br}$  that intersects all discs of the top right group, or (ii) every rectangle with its top left corner on  $c_{tl}$  and its bottom right corner on  $c_{br}$  intersects all discs of the top right group (as in Fig. 3(a)). Both cases are easy to handle, so assume that the intersection of the projects of  $c_{tl}$  and  $c_{br}$  is not empty on the bottom left chain (as in Fig. 3(a)). Assume it consists of  $m$  circle segments. We compute the unrestricted optimal solution for the two circle segments, see Fig. 3(b), and test whether it is valid in  $O(m)$  time. If it is not valid, then the optimal rectangle must have a corner on the bottom left chain as well. In this case, we have three chains with a point on them, and a corner on the bottom left chain defines a rectangle immediately which we must check against the top right chain. Hence, we can find the optimal solution by walking over the  $O(m)$  parts of the bottom left chain from right to left and maintain how the top right corner of the rectangle moves with respect to the top right chain. In the process, we determine the smallest valid rectangle.



**Fig. 4.** (a) The smallest enclosing circle of a set of points in the plane. (b) The largest SEC of a set of imprecise points. (c) The smallest SEC of a set of imprecise points.

Because we go over the circle segments of the bottom right chain from left to right, the projections move in only one direction on the other two chains. Hence, the total complexity of the circle segments we encounter among all cases cannot be more than  $O(n)$  for one circle segment of the top left chain.

**Theorem 3.** *Given a set of  $n$  discs, the problem of choosing a point in each disc such that the axis-aligned bounding box of the resulting point set is as small as possible can be solved in  $O(n^2)$  time.*

### 3. Smallest enclosing circle

We move on to another relatively simple problem. Given a set of points  $P$ , the smallest enclosing circle (SEC) is the smallest circle that contains  $P$ , see Fig. 4(a). When we are given a set  $\mathcal{L}$  of imprecise points, we want to place a point in each region such that the SEC of the resulting point set is as large or as small as possible, see Figs. 4(b) and 4(c).

#### 3.1. Largest possible SEC

##### 3.1.1. Squares

The largest smallest enclosing circle of a set of squares (or constant size convex polygons) can be computed by first computing the smallest enclosing circle of the set of corners of all squares, using an existing SEC algorithm in  $O(n)$  time (e.g. [14]). If the three points that determine this circle belong to different squares, we are done. Otherwise, there is one square of which multiple corners contribute to the smallest enclosing circle, and we know that this square has to contribute to the optimal solution. So we just try all corners of this square and compute the smallest enclosing circle of this single point and all other corners of the other squares.

If the point  $p$  we chose does not lie on the resulting circle, then we clearly chose the wrong point. If  $p$  does lie on the circle, we do not yet know whether we chose the right point, but we proceed by computing the largest smallest circle through  $p$  of the remaining squares in the same way. Since a square has four corners, and at most three points can define a circle, we inspect at most  $4^3$  possible solutions. We report the largest among these.

**Theorem 4.** *Given a set of  $n$  squares, the problem of choosing a point in each square such that the smallest enclosing circle of the resulting point set is as large as possible can be solved in  $O(n)$  time.*

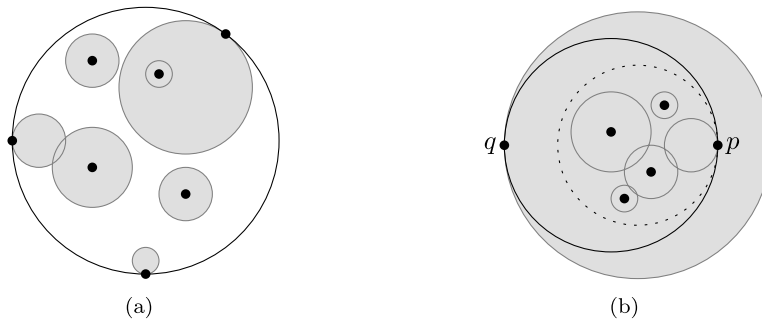
##### 3.1.2. Discs

To compute the largest possible smallest enclosing circle of a set of discs, we observe that there are only two possibilities. Either the largest SEC contains all discs, or it does not, see Fig. 5. If it does, then the largest SEC is just the smallest circle containing a set of discs, which can be computed in  $O(n)$  time [15]. If it does not, this means that there must be one disc among the input discs that contains all other discs. In this case, the largest SEC is determined by the point of all other discs closest to the boundary of this disc, and the point on this disc furthest away from it. This case can clearly also be solved in linear time.

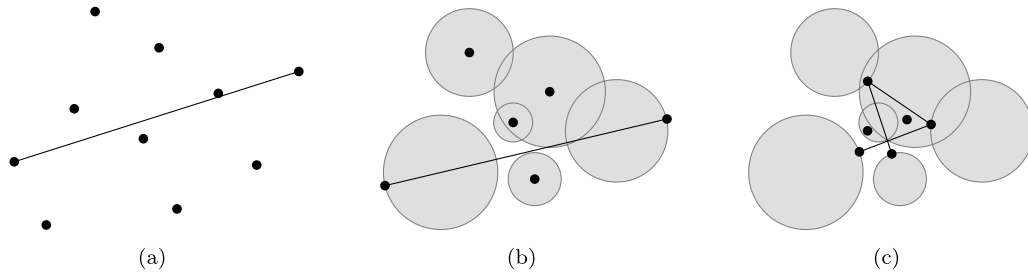
**Theorem 5.** *Given a set of  $n$  discs, the problem of choosing a point in each disc such that the smallest enclosing circle of the resulting point set is as large as possible can be solved in  $O(n)$  time.*

#### 3.2. Smallest possible SEC

The smallest possible SEC for a set of imprecise points is the smallest circle that intersects all regions. This is also called the *intersection radius* of a set of regions. When the regions are squares (or other convex polygons), it can be computed in linear time [8].



**Fig. 5.** The LSEC (black) of a set of discs. (a) All discs are completely within the LSEC. (b) There is one disc containing all others.



**Fig. 6.** The diameter of a set of points in the plane. (b) The largest possible diameter of a set of imprecise points. (c) The smallest possible diameter of a set of imprecise points, determined by three pairs simultaneously.

### 3.2.1. Discs

When the points are modelled as discs, the problem of finding the smallest SEC is LP-type [4]. An LP-type problem is defined on a set of objects  $H$  and a function  $w : 2^H \rightarrow W$ , where  $W$  is some totally ordered set of possible values. The goal is to compute  $w(H)$ . To be LP-type, two axioms must hold:

$$F \subseteq G \subseteq H \implies w(F) \leq w(G)$$

and

$$F \subseteq G \subseteq H, h \in H, w(G) = w(F) < w(F \cup \{h\}) \implies w(G) < w(G \cup \{h\}).$$

LP-type problems can be solved in linear time using a generic algorithm [13]. In our case,  $H = \mathcal{L}$  is the set of discs, and  $w$  gives the radius of the smallest SEC. The first axiom clearly holds. For the second axiom, we are given  $F$  and  $G$ : two collections of discs with the same radius of the smallest SEC. Since  $F \subseteq G$ , they must have the same SEC  $C$ , since the SEC is uniquely defined. Then, we are given a disc  $h$ . Since  $w(F) < w(F \cup \{h\})$ ,  $h$  must lie fully outside  $C$ . But now, clearly,  $w(G) < w(G \cup \{h\})$  as well.

**Theorem 6.** Given a set of  $n$  discs, the problem of choosing a point in each disc such that the smallest enclosing circle of the resulting point set is as small as possible can be solved in  $O(n)$  time.

## 4. Diameter

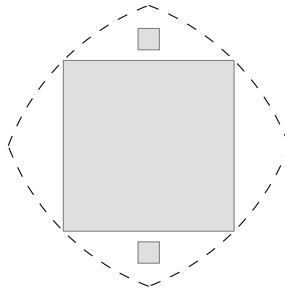
Given a set of points  $P$ , the diameter is the largest distance between any pair of points in  $P$ , see Fig. 6(a). When the points are imprecise, we are given a set  $\mathcal{L}$  of regions, and we want to place a point in each region such that the diameter of the resulting point set is as large or as small as possible, see Figs. 6(b) and 6(c).

### 4.1. Largest possible diameter

The largest possible diameter is formed by the pair of points among the input regions that are furthest away from each other, unless they belong to the same region.

#### 4.1.1. Squares

When the points are modelled as squares, the two points forming the largest diameter must be among the corners of the squares. This means we can just compute the diameter of the set of all corners using a conventional diameter algorithm in  $O(n \log n)$  time. If the two points found belong to different squares, we are done.



**Fig. 7.** The largest possible diameter is formed by two corners of the small squares, although the largest diameter among all corners is a diagonal of the big square.

Otherwise, they are diagonally opposite points of one square  $s$ , and there are two possibilities. Either the largest diameter is formed by one corner of  $s$  and one corner of another square, or the largest diameter is formed by two points among the other squares, see Fig. 7. To handle the first case, we can simply check each corner of  $s$  with all other corners in  $O(n)$  time. To handle the second case, we remove  $s$  and compute the diameter of the corners of all other squares. They must lie on different squares, which is shown by the following lemma.

**Lemma 2.** *If the largest diameter of the corners of all squares is formed by two corners of one square  $s$ , and there are two points  $p$  and  $q$  in the remaining squares that are further away from each other than from  $s$ , then  $p$  and  $q$  must belong to two different squares.*

**Proof.** Both points must lie outside  $s$ , because if not we could take a corner of  $s$  instead and get a larger diameter. Furthermore, they must lie in two different triangle-like regions as in the figure. Now, if they belonged to the same square, this square would have to contain at least one corner of  $s$ , which means that it must have a corner outside  $s$ , and that corner with the opposite corner of  $s$  would have given a larger diameter than  $s$  alone, a contradiction.  $\square$

**Theorem 7.** *Given a set of  $n$  squares, the problem of choosing a point in each square such that the diameter of the resulting point set is as large as possible can be solved in  $O(n \log n)$  time.*

#### 4.1.2. Discs

Compute the convex hull of the set of discs [18] in  $O(n \log n)$  time, and use rotating calipers to find the diameter. We will find two points that are on some disc. If these discs are different, we are done. Otherwise, there is again one big disc that contains all others, like in the largest SEC problem, and we must find the point closest to the boundary of this disc again.

**Theorem 8.** *Given a set of  $n$  discs, the problem of choosing a point in each disc such that the diameter of the resulting point set is as large as possible can be solved in  $O(n \log n)$  time.*

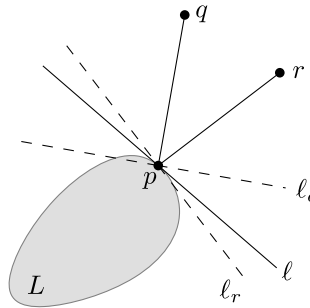
#### 4.2. Smallest possible diameter

Computing the smallest possible diameter  $d$  of  $\mathcal{L}$  is a difficult problem. The reason is that it can be determined by multiple pairs of points simultaneously, as could already be seen in Fig. 6(c). Moving any of the four points involved would increase the distance between at least one pair of them. In general, all  $n$  points could be involved in such a construction, where none of the points can be moved without increasing the diameter. Note that these situations are not degenerate. In the case of discs, it means that we have to resort to approximation algorithms, but we will show that in the case of squares, an exact solution is possible.

For any subset  $\mathcal{L}' \subset \mathcal{L}$ , let  $d'$  be the value of the smallest possible diameter of  $\mathcal{L}'$  and let  $P'$  be the set of points that achieves it. There will be some pairs of points in  $P'$  that have distance exactly  $d'$  to each other. These pairs define a graph on  $P'$ . We remove any edges in this graph that can be removed by simply moving one point of  $P'$  (alternatively, we could say we have an edge between two points only if the distance between them is exactly  $d'$  in any optimal placement of the points). From now on, we refer to this graph as a *star*.

We can make some observations about the nature of such a star (which also motivate its name). If  $\mathcal{L}$  is in general position, the graph will be connected, and every vertex in the graph will have degree 1 or 2, which implies that the graph is actually a path or a cycle.<sup>1</sup> Furthermore, all edges of this graph have the same length  $d'$ . Since this is the diameter of  $P'$ ,

<sup>1</sup> The assumption of general position is not restrictive for the algorithm: if the graph is more complex, then the algorithm will simply return a subgraph of it that is a path or a cycle.



**Fig. 8.** A point  $p$  on the optimal star.  $L$  cannot intersect the section of the plane bounded by  $\ell_q$  and  $\ell_r$  where  $q$  and  $r$  lie, otherwise  $p$  could move closer to  $q$  and  $r$ . Therefore  $p$  is extreme in  $L$ .

no two points can be more than  $d'$  away from each other. Therefore all edges must intersect each other, and the path makes an angle of at most  $60^\circ$  at each degree 2 vertex. We also call such a vertex a *bend* of the star. A bend is a point that is in balance between its two neighbours, it could move closer to one neighbour but only by moving further from the other neighbour. The bends are what make the problem hard. Some examples of stars can be seen in Figs. 6(c), 9(b) and 12(a).

Ultimately, we want to compute the star of  $\mathcal{L}$ , which we call the *optimal star*. We first make some observations about this star that are true for any convex regions. Then we show how to solve the problem efficiently for square regions, and we present approximation algorithms for circular regions.

**Observation 1.** Let  $p$  be a point on the optimal star, and let  $q$  be an adjacent point on the star. Let  $\ell$  be the line through  $p$  perpendicular to  $\overline{pq}$ . Then no region is entirely on the other side of  $\ell$  than where  $q$  is.

The reason why this observation is true, is that such a region would be more than  $d$  away from  $q$ , which contradicts optimality of the star. With this observation, we can introduce some more definitions and facts. Let  $L \in \mathcal{L}$  be a region. We call  $L$  an *extreme* region if there exists a line  $\ell$  that has the interior of  $L$  completely on one side, but no other region of  $\mathcal{L}$  has its interior completely on the same side of  $\ell$ . We call a point  $p \in L$  an *extreme placement* if such a line exists that goes through  $p$ .

**Observation 2.** All points of the optimal star must be on extreme placements in extreme regions.

**Proof.** Let  $L \in \mathcal{L}$  be a region and  $p \in L$  a vertex of the optimal star. Consider the vertices  $q$  and  $r$  of the star that are adjacent to  $p$  (possibly there is only one), and consider the lines  $\ell_q$  and  $\ell_r$  through  $p$  that are perpendicular to  $pq$  and  $pr$ . Fig. 8 illustrates the situation. Because  $p$  is in its optimal position, there is no point in  $L$  that is closer to both  $q$  and  $r$  than  $p$  is. Since  $L$  is convex, this means there exists a line through  $p$  that has the interior of  $L$  completely on one side. By Observation 1, there are no other regions completely on the other side of  $\ell_q$  than  $q$ , and no regions completely on the other side of  $\ell_r$  than  $r$ . Therefore, there are no other regions completely on the same side of  $\ell$  as  $L$ .  $\square$

In almost every direction, there is exactly one extreme region, except for the critical directions where a line is tangent to two regions simultaneously. If we rotate through all directions, we find the *critical sequence* of the regions, as introduced by Rappaport for a set of line segments [19]. When the regions are circles or squares, this sequence can be computed in  $O(n \log n)$  time [17].

#### 4.2.1. Squares

When the points are modelled as squares, we can solve the smallest diameter problem in  $O(n \log n)$  time. We first investigate where the vertices of the optimal star could be. Among the extreme squares, there are only four with infinitely many extreme placements, being the squares with the topmost bottom side, the bottommost top side, the leftmost right side and the rightmost left side. We call these squares *axis-extreme*. The other extreme squares can only have an extreme placement at one corner. These placements form four *chains*: the top left chain connects all bottom right extreme placements, etc., see Fig. 9(a). Note that these chains are convex. The extreme squares and chains can be computed in  $O(n \log n)$  time [12].

Assuming the squares are in general position, the optimal star cannot have bends at corners of squares. If there would be a bend  $p$  at a corner, there would be at least one neighbour  $q$  of  $p$  such that  $p$  would not be able to move closer to  $q$ , even if it had no other neighbour. This means  $p$  is not in balance, making this in fact a degenerate case and not a real bend. Therefore, the only interesting bends occur at axis-extreme squares. Together with the fact that all edges of a star must intersect each other, this means that the optimal star can have at most two bends, as in Fig. 9(b). This implies that we can find it efficiently, as we will now show.



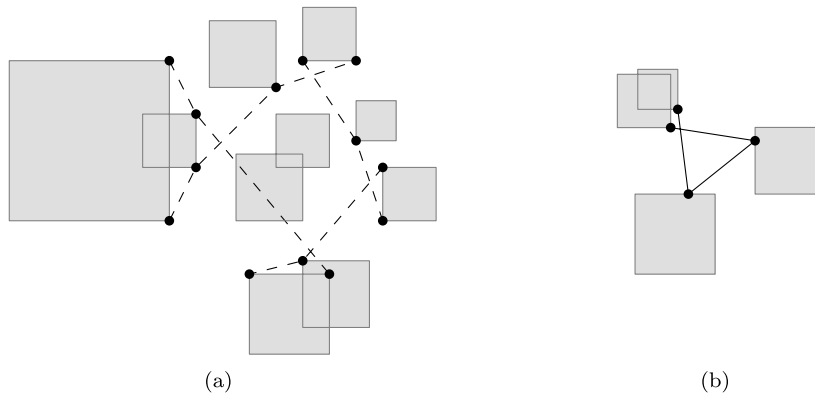


Fig. 9. (a) Four extreme squares and four chains. (b) A star with two bends.

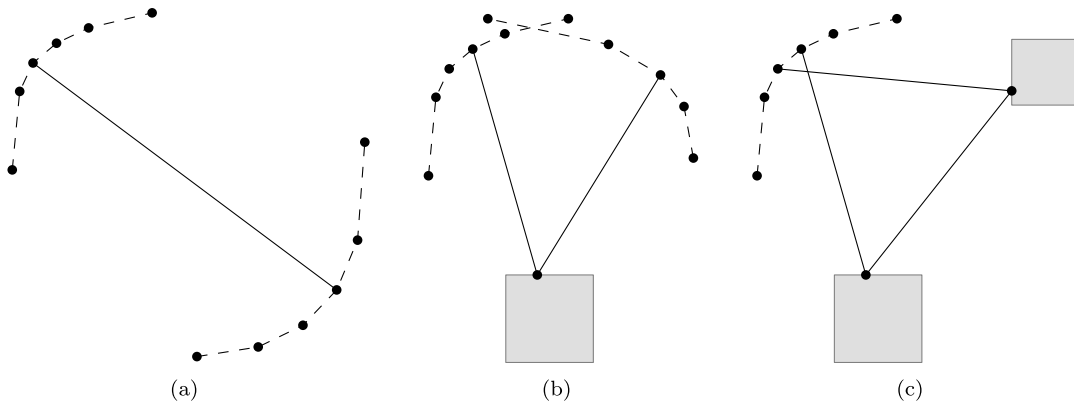


Fig. 10. (a) The optimal star is a direct connection between two chain vertices. (b) The optimal star has one bend in an axis-extreme square. (c) The optimal star has two bends in consecutive axis-extreme squares.

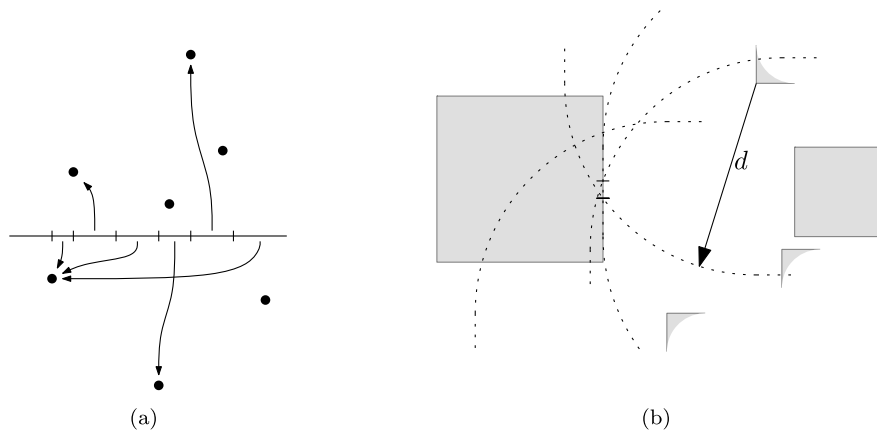
**Lemma 3.** *We can find the optimal star by computing the star of every subset of four squares, of which two are axis-extreme, and reporting the largest among these.*

**Proof.** The algorithm returns a star for subset  $\mathcal{L}' \subset \mathcal{L}$ . Let this star have diameter  $d'$ . Clearly, the diameter  $d$  of  $\mathcal{L}$  must then be at least as large as  $d'$ . On the other hand, it cannot be larger, because the optimal star has only four vertices, of which two are on axis-extreme squares, so this set of squares must have been considered too. Therefore,  $d = d'$ .  $\square$

As an immediate result, we can solve the problem in  $O(n^2)$  time by enumerating all these sets of squares and computing their stars. However, we will now show that after precomputing the chains of possible extreme placements we can also find the optimal star in linear time, by using a careful case analysis and using the structure of the chains. Together with the computation of these chains, this yields an  $O(n \log n)$  algorithm. In overview, for every placement of a point on an axis-extreme square, there is one vertex on one of the chains that is furthest away from it, and this determines the best possible diameter in this case. As the axis-extreme point moves over its edge, this furthest vertex can move only in restricted ways, which saves us a linear factor.

Assume we have computed in  $O(n \log n)$  time the four axis-extreme squares and the four chains of extreme squares, as in Fig. 9(a). Since the optimal star has at most two bends and its edges must intersect each other, it can only be of three different types. It may be a single connection between two squares without any bends, or a star with one bend on an axis-extreme square and two endpoints on the opposite chains, or a star with two bends on consecutive axis-extreme squares and two endpoints on the same opposite chain, see Fig. 10. We will try all cases and all symmetric possibilities within a case, and compute the largest possible valid star in each case in linear time. The largest among these must be the optimal star.

**Interval division.** In order to solve the different cases, we will need a simple structure that divides a line into intervals. Given a set of points  $P$ , we define a function  $f(x)$  as follows. Let  $L_P(x) \subset P$  be the set of those points in  $P$  with an  $x$ -coordinate at most  $x$ . Then  $f(x)$  is the furthest point in  $L_P(x)$  from the point  $(x, 0)$ . We divide the  $x$ -axis into intervals



**Fig. 11.** (a) The interval division of a set of 7 points. (b) The leftmost axis-extreme points must be placed within the dotted circles.

where  $f$  is constant, see Fig. 11(a). There will be an unbounded interval to the left of the leftmost point where  $f$  is not defined.

**Lemma 4.** *The interval division has linear complexity and can be computed in  $O(n \log n)$  time.*

**Proof.** We can compute the interval division incrementally by sorting the points by increasing  $x$ -coordinate, and inserting them in that order. When we know the interval division of the  $x$ -axis with respect to the points  $\{p_1, \dots, p_i\}$ , and we insert the next point  $p_{i+1}$  with  $x$ -coordinate  $x_{i+1}$ , we observe that at most one new interval can be created and that this new interval must start at  $x_{i+1}$ . The new interval may (partially) overlap any number of existing intervals, which we can find by scanning to the right until the new interval ends. With each new point, only a single new interval is created, therefore only a linear number of intervals is created in total. This means that also only a linear number can be overwritten during the scans, and the total time spent is linear.  $\square$

We define the interval division of any directed line by transforming that line onto the  $x$ -axis.

**Case 1: no bends.** When the optimal star contains no bends, it is a direct connection between two regions, see Fig. 10(a). This can either be a horizontal or vertical connection between two opposite axis-extreme squares, or a diagonal connection between two vertices of opposite chains. The former case can be computed in constant time. In the latter case we need to ensure that we only consider pairs with the right slope: positive for a connection between the bottom left and top right chains, negative for a connection between the top left and bottom right chains. This can be computed by a simple variation to the conventional diameter algorithm using rotating calipers [22] in linear time.

**Case 2: one bend.** When the optimal star has exactly one bend, this bend is on an axis-extreme square, say the bottommost square, see Fig. 10(b). The start and end points of the star must be vertices of the top left and top right chains. To find the largest star of this type, we must find the point  $p$  on the bottommost square such that the distance to the furthest point on both chains is minimised. However, we must only consider points of the top left chain if they are to the left of  $p$ , and only points of the top right chain if they are to the right of  $p$ .

Let  $l$  be the horizontal line through the top side of the bottommost axis-extreme square. We compute the interval division of the directed line  $l$  from left to right with respect to the set of points that form the top left chain, and we also compute the interval division of  $l$  directed from right to left with respect to the points in the top right chain. We can now balance the furthest points on both chains in linear time.

**Case 3: two bends.** When the optimal star has two bends, these bends must occur at consecutive extreme squares, say the bottommost and rightmost ones, see Fig. 10(c). The start and end points are then vertices of the top left chain. We must now find the point  $p$  on the bottom square and the point  $q$  on the right square that minimise the largest among the distance from  $p$  to the furthest vertex of the chain to the left of  $p$ , from  $q$  to the furthest vertex of the chain above  $q$ , and from  $p$  to  $q$ .

To find these optimal positions, we again compute the interval division of the line from left to right through the top side of the bottommost axis-extreme square with respect to the points on the top left chain, and we similarly partition the line from top to bottom through the left side of the rightmost axis-extreme square with respect to the same point set. Now we place  $p$  at its locally optimal position, and  $q$  too. The furthest distance must occur between  $p$  and  $q$ , otherwise the optimal star was not of this type. Now start moving the points towards each other, keeping their distances to the furthest point on the chain equal. They both move only in one direction, so we can find the optimal location in linear time again.

*Placing the points.* We have now computed the value  $d$  of the smallest possible diameter. If needed, we can also compute a placement of the points in their regions that realises this diameter. We first compute valid placements of the four axis-extreme points, and then observe that all other points should be placed ‘as far inward’ as possible.

To compute a valid placement in an axis-extreme square, note the following. If we find a placement such that any star that includes this point has at most length  $d$ , then this placement is valid for a global solution of diameter  $d$ . This means we can check for all possible stars in which interval the point is allowed to lie if that star must be at most  $d$  long, and then place the point somewhere in the intersection of all these intervals.

To determine the intervals of an axis-extreme square, say the bottommost, where a point could be placed that still allows for a solution of length  $d$ , we compute for every square the interval that is at most  $d$  away from that square. The intersection of these intervals gives an interval where a point can still be placed that is at most  $d$  away from any other square, see Fig. 11(b). After that, we must still place the axis-extreme points in such a way that they are at most  $d$  away from each other. However, we already know that this is possible and since there are only four axis-extreme points we can place them in constant time. Any placement within the precomputed intervals will be fine with respect to the  $n - 4$  other squares.

For the rest of the squares, if a point is to the left of the vertical lines through the topmost and bottommost axis-extreme points, moving it to the right can only decrease the diameter, and a similar statement holds for the other extremes. Because the regions are squares, every point will end either in a corner of its region or somewhere in the middle of the whole construction.

**Theorem 9.** *Given a set of  $n$  squares, the problem of choosing a point in each square such that the diameter of the resulting point set is as small as possible can be solved in  $O(n \log n)$  time.*

#### 4.2.2. Discs

When the points are modelled as discs, stars can have up to  $n$  bends, see Fig. 12(a). This leads to algebraic difficulties: even if we knew the combinatorial structure of the optimal star, computing the positions where the points are in balance exactly would not be possible.

We can make some observations about the combinatorial structure. We can still define the extreme discs, that is, discs that have a tangent line with no other disc completely on the same side. These discs form a chain of arcs with this property, see Fig. 12(b). Any bend in the star still needs to be on such an extreme arc. However, it is possible that all discs have extreme arcs.

Since we cannot compute the optimum efficiently, we provide two approximation algorithms. A factor  $\frac{2}{3}\sqrt{3} \approx 1.15$  approximation can be computed in linear time by computing the smallest enclosing circle of the imprecise points. We can also compute a  $(1 + \varepsilon)$ -approximation of the smallest diameter in  $O(n^{c\varepsilon^{-1/2}})$  time.

*Constant factor approximation.* The approximation is based on the fact that for any given point set  $P$ , the diameter  $d$  of  $P$  is at least  $\sqrt{3}$  times the radius  $r$  of the smallest enclosing circle of  $P$ . This is because of the three points that define the SEC, there must be two that are at least as far apart as if they lay on an equilateral triangle. Also,  $d$  can at most be  $2r$ .

Now let  $\mathcal{L}$  be a set of imprecise points, let  $P$  be a set of points that realises the smallest possible diameter and let  $Q$  be a set of points that realises the smallest possible SEC.

**Lemma 5.** *The diameter of  $Q$  is at most  $\frac{2}{3}\sqrt{3}$  larger than the diameter of  $P$ .*

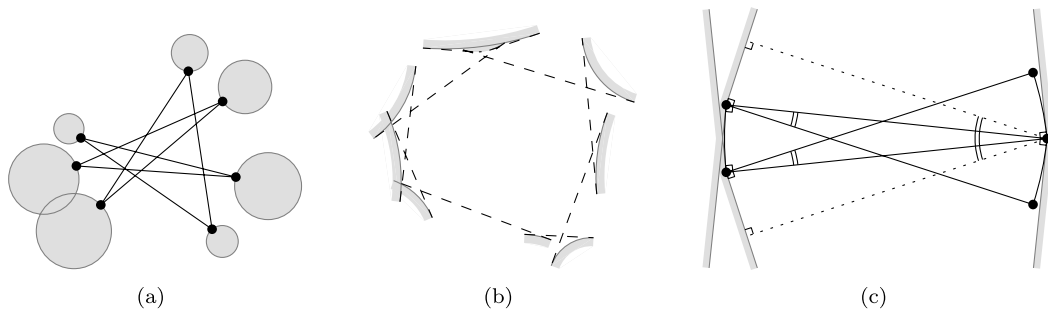
**Proof.** Let  $d_P$  be the diameter of  $P$ ,  $d_Q$  the diameter of  $Q$ ,  $r_P$  the radius of the SEC of  $P$  and  $r_Q$  the radius of the SEC of  $Q$ . Then we have:

$$d_Q \leq 2r_Q \leq 2r_P = \frac{2}{3}\sqrt{3}(\sqrt{3}r_P) \leq \frac{2}{3}\sqrt{3}d_P. \quad \square$$

Since we can compute the smallest SEC in linear time, we can approximate the smallest diameter within a factor  $\frac{2}{3}\sqrt{3}$  in linear time as well.

*Approximation scheme.* We can compute a  $(1 + \varepsilon)$ -approximation in  $O(n^{3\pi\varepsilon^{-1/2}})$  time. The idea of this algorithm is to consider only stars of at most  $k$  bends, for  $k$  chosen suitably. We compute a subset  $\mathcal{L}' \subset \mathcal{L}$  for which we can show that the optimal solution  $d'$  is at most a factor  $(1 - \frac{1}{2}\varepsilon)$  shorter than the real optimum  $d$ . Therefore, we get a  $1/(1 - \frac{1}{2}\varepsilon) = (1 + \frac{1}{2}\varepsilon + o(\varepsilon^2))$ -approximation, which is an  $(1 + \varepsilon)$ -approximation if  $\varepsilon \leq 1$ .

**Lemma 6.** *Suppose the optimal solution is given by a star of at least  $k$  bends. Then there exists a star with only one bend that approximates it within a factor of  $1 - O(k^{-2})$ .*



**Fig. 12.** (a) A cyclic star that visits seven regions. (b) Circular arcs that are extreme in some direction. (c) Three consecutive angles of at most  $\alpha$  give a diameter of  $\cos \alpha$ .

**Proof.** Suppose that the optimal diameter is 1. The sum of the angles that the star makes in the bends is at most  $\pi$ . That means that there are three consecutive bends  $a$ ,  $b$  and  $c$  somewhere that together make an angle of at most  $\alpha = \frac{3\pi}{k}$ . Therefore the individual angles are also at most  $\alpha$ , see Fig. 12(c).

The region  $L \subset \mathcal{L}$  that supplied the point  $b$  is convex and completely outside the wedge that is formed by the halfline from  $b$  perpendicular to  $\overline{ab}$  in the direction of  $c$ , and the halfline from  $b$  perpendicular to  $\overline{bc}$  in the direction of  $a$ , otherwise  $b$  would not be in its optimal position. In the same way, the regions of  $a$  and  $c$  are also convex and behind the wedges formed by their two neighbours on the star, unless one of them is the endpoint of the star and the wedge degenerates to a line.

This means that in the worst case the regions of  $a$  and  $c$  are arbitrarily close to the halflines that come nearest to  $b$ , and in that case the best possible diameter of  $a$ ,  $b$  and  $c$  would be  $\cos \alpha$ , as denoted by the dotted lines in Fig. 12(c). This is more than  $1 - \frac{1}{2}\alpha^2 = 1 - \frac{9\pi^2}{2k^2}$ .  $\square$

To get a  $(1 - \frac{1}{2}\varepsilon)$ -approximation we take  $k = 3\pi\varepsilon^{-1/2}$  (and at least 3) and compute all chains of length at most  $k$  in  $O(n^k)$  time. If the optimal star has at most  $k$  bends, we will find it. If not, then by Lemma 6, there exists a good approximate star of length 3, which we will find.

**Theorem 10.** Given a set of  $n$  discs, the problem of choosing a point in each disc such that the diameter of the resulting point set is as small as possible can be approximated within a factor  $(1 + \varepsilon)$  in  $O(n^{c\varepsilon^{-1/2}})$  time, where  $c = 3\pi$ .

## 5. Width

Given a set of points  $P$ , the width is the smallest distance between any pair of parallel lines that contains  $P$ , see Fig. 13(a). Examples of the imprecise case are in Figs. 13(b) and 13(c).

### 5.1. Largest possible width

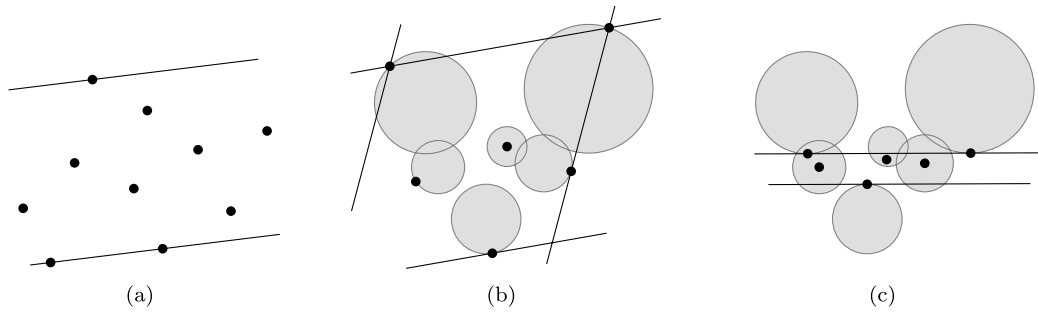
This problem seems to be hard. When the points are modelled as line segments, it is even NP-hard.

#### 5.1.1. Line segments

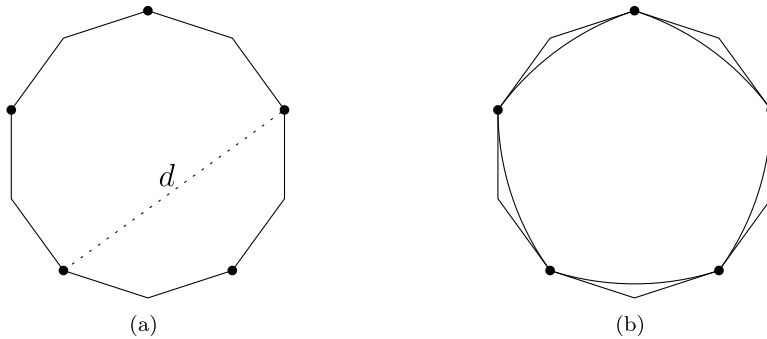
Computing the largest possible width of a set of imprecise points modelled as arbitrarily oriented line segments is NP-hard. We prove this by reduction from SAT. The construction is similar to the one used to prove NP-hardness of computing the largest possible convex hull of a set of line segments [11], but the nature of the width measure requires some new ideas.

Given a SAT instance, let  $k$  be odd and larger than the number of clauses and variables in the instance together. We base the construction on a regular  $2k$ -gon. We place  $k$  precise points distributed evenly on the vertices of the  $2k$ -gon, see Fig. 14(a). Let the furthest distance between two of these precise points be  $d$ . The imprecise points that we will place later will all be completely within the  $2k$ -gon. This ensures that the largest width can at most be  $d$ , because the width of the  $2k$ -gon itself is  $d$ . We will make a construction such that a width of  $d$  arises if and only if the SAT instance can be satisfied.

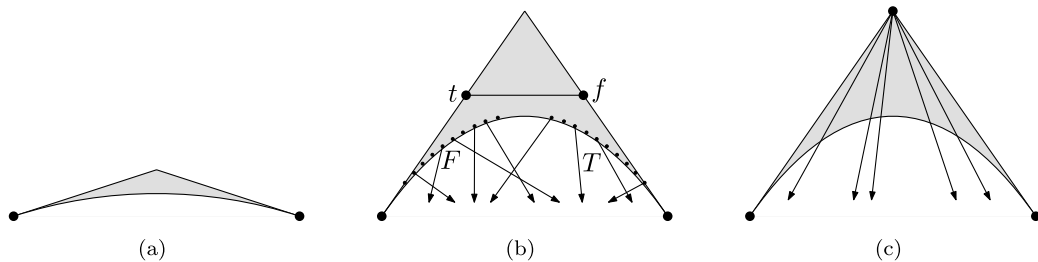
Now, for all  $k$  precise points we draw an arc with that point as centre between its two opposite points. These arcs together form a curve of constant width, see Fig. 14(b). This is the smallest possible region within the  $2k$ -gon that contains the  $k$  precise points and has width  $d$ . As a consequence, a solution of width  $d$  is possible if and only if a solution the imprecise points can be placed in such a way that the convex hull of the points completely contains this curve of constant width. There are  $k$  parts between the  $2k$ -gon and the curve of constant width, see Fig. 15(a), which we will use to construct variables and clauses.



**Fig. 13.** (a) The width of a set of points in the plane. (b) The largest possible width of a set of imprecise points, reached at two different locations simultaneously. (c) The smallest possible width of a set of imprecise points.



**Fig. 14.** (a) A regular  $2k$ -gon with width  $d$ . (b) A curve of constant width, formed by  $k$  arcs.



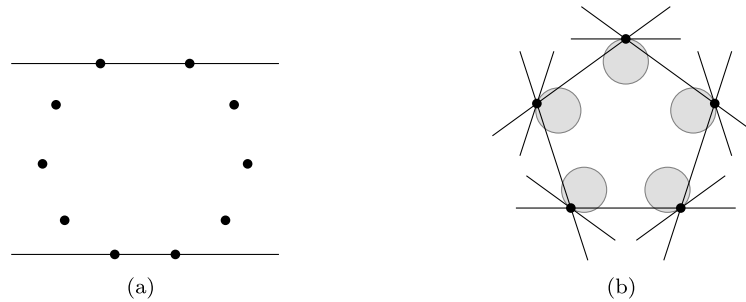
**Fig. 15.** (a) A part between the  $2k$ -gon and the curve of constant width. (b) A variable inside a part. (c) A clause inside a part.

**Lemma 7.** A solution has width  $d$  if and only if the convex hull of the placed points completely contains the curve of constant width (and stays within the  $2k$ -gon).

**Proof.** If the convex hull of the points completely contains the curve of constant width, its width must be at least  $d$ . On the other hand, it is still completely contained in the  $2k$ -gon, which also has width  $d$ , so its width is at most  $d$ . Therefore, it is  $d$ .

If the convex hull of the points does not completely contain the curve of constant width, there is some point  $p$  in the interior of the curve of constant width that is not inside the convex hull. This means that there is a line  $l$  through  $p$  that does not intersect the convex hull, and therefore does not intersect the  $k$ -gon formed by the  $k$  precise points. Because  $k$  is odd, there is a line parallel to  $l$  through one of the  $k$  precise points that does not intersect the  $2k$ -gon anywhere else, and since this point is part of the curve of constant width the distance between these two lines is less than  $d$ . Since the convex hull is completely between them, the width of the placed points is also less than  $d$ .  $\square$

The rest of the reduction is exactly the same as in [11], and we refer the reader to that paper for the full details. For completeness, we sketch the argument briefly here. For each Boolean variable  $b$  in the SAT instance, we add the configuration of Fig. 15(b), consisting of two points  $t$  and  $f$  and two sets of points  $T$  and  $F$ , inside an empty part. We can only use one of the points  $t$  and  $f$ , so if the convex hull of the resulting point set must contain the circular arc we need either  $t$  and all points in  $T$  (representing the value `true` of this variable) or  $f$  and all points in  $F$  (representing `false`). For each clause we add a single point  $s$  at the top of an empty part, see Fig. 15(c). We include a line segment (an imprecise point)



**Fig. 16.** (a) A set of  $n$  points, such that the width changes when any of them is removed. (b) Many triples of points define the width simultaneously.

between  $s$  and every variable in this clause. If the variable occurs normally, the other endpoint is in the  $T$ -group, and if it is negated the other endpoint is in the  $F$ -group. Since we can use only one endpoint of every segment, we can only place a point at  $s$  if at least one of the variables of the clause is in the right state. Therefore, we can make a point set whose convex hull contains the curve of constant width if and only if the SAT instance is satisfiable, and together with Lemma 7 this implies NP-hardness of the width problem.

**Theorem 11.** *Given a set of  $n$  arbitrarily oriented line segments, the problem of choosing a point on each segment such that the width of the resulting point set is as large as possible is NP-hard. The decision version of the problem is NP-complete.*

### 5.1.2. Squares or discs

The width of a set of points has the property that there can be points that do not contribute to the width, but are needed to make the width valid, see Fig. 16(a). Removing any point here would result in a smaller width. In an imprecise context, this means that the placement of all points is important, and we cannot look only at subsets of the regions. Furthermore, it can happen that many points are simultaneously involved in the optimal width, see Fig. 16(b), in a similar way as the stars in the smallest diameter problem. The status of the largest width problem for imprecise points modelled as squares or discs is open.

### 5.2. Smallest possible width

The smallest width of a set of imprecise points can be computed efficiently. For squares (or any polygonal regions), the problem can be solved in  $O(n \log n)$  time [20].

For discs, we compute the critical sequence of extreme placements again, see Fig. 12(b), in  $O(n \log n)$  time [17]. The smallest width must have a critical line on one side, so we can find it by using rotating calipers.

**Theorem 12.** *Given a set of  $n$  discs, the problem of choosing a point in each disc such that the width of the resulting point set is as small as possible can be solved in  $O(n \log n)$  time.*

## 6. Closest pair

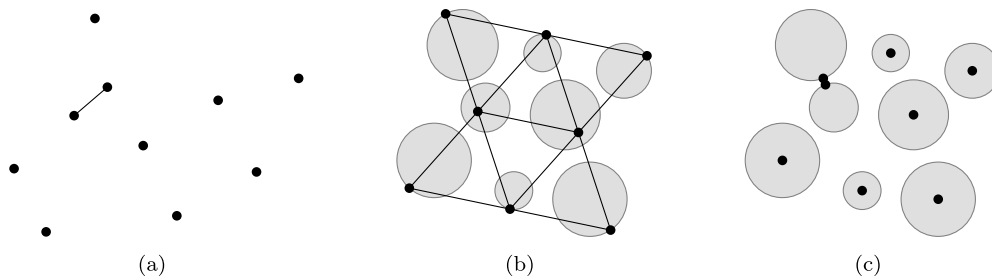
We conclude with another related problem. Given a set of points  $P$ , the closest pair is the pair of points with the smallest distance between them among all pairs of points in  $P$ , see Fig. 17(a). In an imprecise context, we are given a set  $\mathcal{L}$  of regions, and we want to place a point in each region such that the closest pair of the resulting point set is as large or as small as possible, see Figs. 17(b) and 17(c).

### 6.1. Largest possible closest pair

Maximising the closest pair distance means that we need to place all points in such a way that their mutual distances are balanced. This is known as *spreading points*. Fiala et al. [6] proved this to be NP-hard, both in the square and the circle model. Cabello [3] gives a constant factor approximation algorithm.

### 6.2. Smallest possible closest pair

To minimise the closest pair distance, we just need to find the two regions that are closest to each other and place their points as close together as possible. First, we check whether any two regions intersect each other in  $O(n \log n)$  time by a standard sweep. If there is any intersection, the smallest possible closest pair has distance 0. Otherwise, we compute the Voronoi diagram of a set of convex objects [23] in  $O(n \log n)$  time. The closest pair is determined by objects in adjacent cells, so we can find them efficiently.



**Fig. 17.** (a) The closest pair of a set of points in the plane. (b) The largest possible closest pair of a set of imprecise points, reached by many pairs simultaneously. (c) The smallest possible closest pair of a set of imprecise points.

**Theorem 13.** *Given a set of  $n$  squares or discs, the problem of choosing a point in each square or disc such that the closest pair of the resulting point set is as small as possible can be solved in  $O(n \log n)$  time.*

## 7. Conclusions

We have given a structured overview of the imprecise variants of several basic geometric measures, reusing known results from various other contexts and designing efficient algorithms for the remaining cases. Most problems can be computed efficiently, while some are NP-hard. One remaining open problem is that of computing the largest possible width for the square or disc model. Also, the 3D versions of most problems are still open.

## References

- [1] M. Abellanas, F. Hurtado, P.A. Ramos, Structural tolerance Delaunay triangulation, *Inform. Process. Lett.* 71 (1999) 221–227.
- [2] I. Averbakh, S. Bereg, Facility location problems with uncertainty on the plane, *Discrete Optimization* 2 (2005) 3–34.
- [3] S. Cabello, Approximation algorithms for spreading points, *J. Algorithms* 62 (2) (2007) 49–73.
- [4] B. Chazelle, J. Matoušek, On linear-time deterministic algorithms for optimization problems in fixed dimension, *J. Algorithms* 21 (1996) 579–597.
- [5] P. Colley, H. Meijer, D. Rappaport, Optimal nearly-similar polygon stabbers of convex polygons, in: *Proc. 6th Canad. Conf. Comput. Geom.*, 1994, pp. 269–274.
- [6] J. Fiala, J. Kratochvíl, A. Proskurowski, Systems of distant representatives, *Discrete Appl. Math.* 145 (2005) 306–316.
- [7] L.J. Guibas, D. Salesin, J. Stolfi, Constructing strongly convex approximate hulls with inaccurate primitives, *Algorithmica* 9 (1993) 534–560.
- [8] S. Jadhav, A. Mukhopadhyay, B.K. Bhattacharya, An optimal algorithm for the intersection radius of a set of convex polygons, *J. Algorithms* 20 (1996) 244–267.
- [9] W.S. Kendall, O. Barndorff-Nielsen, M.C. van Lieshout, *Current Trends in Stochastic Geometry: Likelihood and Computation*, CRC Press, Boca Raton, FL, 1998.
- [10] H. Kruger, Basic measures for imprecise point sets in  $\mathbb{R}^d$ , Master's thesis, Utrecht University, 2008.
- [11] M. Löffler, M. van Kreveld, Approximating largest convex hulls for imprecise points, *J. Discrete Algorithms* 6 (4) (2008) 583–594.
- [12] M. Löffler, M. van Kreveld, Largest and smallest convex hulls for imprecise points, *Algorithmica* (2008), doi:10.1007/s00453-008-9174-2, in press.
- [13] J. Matoušek, M. Sharir, E. Welzl, A subexponential bound for linear programming, *Algorithmica* 16 (1996) 498–516.
- [14] N. Megiddo, Linear-time algorithms for linear programming in  $R^3$  and related problems, *SIAM J. Comput.* 12 (4) (1983) 759–776.
- [15] N. Megiddo, On the ball spanned by balls, *Discrete Comput. Geom.* 4 (1989) 605–610.
- [16] T. Nagai, N. Tokura, Tight error bounds of geometric problems on convex objects with imprecise coordinates, in: *Proc. Jap. Conf. on Discrete and Comput. Geom.*, 2000, pp. 252–263.
- [17] T. Nagai, S. Yasutome, N. Tokura, Convex hull problem with imprecise input its solution, *Systems and Computers in Japan* 30 (3) (1999) 31–42.
- [18] D. Rappaport, A convex hull algorithm for discs applications, *Comput. Geom. Theory Appl.* 1 (3) (1992) 171–181.
- [19] D. Rappaport, Minimum polygon transversals of line segments, *Internat. J. Comput. Geom. Appl.* 5 (1995) 243–256.
- [20] J. Robert, G. Toussaint, *Computational geometry and facility location*, Technical Report SOCS 90.20, McGill Univ., Montreal, PQ, 1990.
- [21] A. Rosenfeld, Fuzzy geometry: An updated overview, *Inform. Sci.* 110 (3–4) (1998) 127–133.
- [22] M.I. Shamos, *Computational geometry*, Ph.D. thesis, Dept. Comput. Sci., Yale Univ., New Haven, CT, 1978.
- [23] C.K. Yap, An  $O(n \log n)$  algorithm for the Voronoi diagram of a set of simple curve segments, *Discrete Comput. Geom.* 2 (1987) 365–393.